# *Cryptography:*
# *Mathematics Trumps Black Magic*

Paul Carr

Western Washington University

## Obligatory Outline Thing

- Crypto 102
- Building a Model
- Useful Tools
- (Vague) Goals
- Crypto 114 (Cryptanalysis)
- Death of a Round Function
- The Punchline
- Rijndael

Q: How many cryptographers does it take to change a light bulb?
A: XIGHCBS

## Block Ciphers

- Purpose of Cryptography: Turn large secrets into small ones
- Main Jargon: cipher, plaintext, ciphertext, key
- $F_D(F_E(x, k_E), k_D) = x$
- Main distinction: Symmetric vs. Asymmetric

**Properties of Symmetric Key Ciphers**: $k_E = k_D$, brute force obfuscation of data by repeated application of basic computer operations, security derived from complexity, key sizes range from 128-256 bits typically

**Properties of Asymmetric (Public) Key Ciphers**: $k_E \neq k_D$, typically based on mathematical “trapdoor" functions, security derived from difficulty of certain math problems, key sizes range from 1024-2048 bits typically

## Block Ciphers

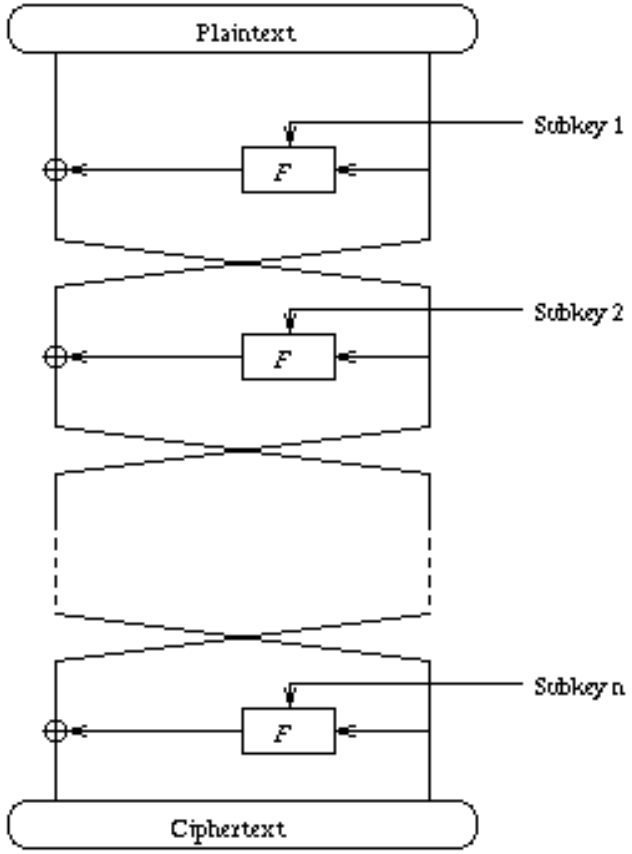
- Purpose of Cryptography: Turn large secrets into small ones
- Main Jargon: cipher, plaintext, ciphertext, key
- $F_D(F_E(x, k_E), k_D) = x$
- Main distinction: Symmetric vs. Asymmetric

**Properties of Symmetric Key Ciphers**: $k_E = k_D$, brute force obfuscation of data by repeated application of basic computer operations, security derived from complexity, key sizes range from 128-256 bits typically, FAST!

**Properties of Asymmetric (Public) Key Ciphers**: $k_E \neq k_D$, typically based on mathematical “trapdoor" functions, security derived from difficulty of certain math problems, key sizes range from 1024-2048 bits typically, SLOW!

# Symmetric Key Block Ciphers

- Structured in “rounds"
- Rounds often identical
- Round keys derived from main key by use of a “key schedule"
- Structure allows the construction of a somewhat simple round function which is applied many times to achieve total plaintext obfuscation

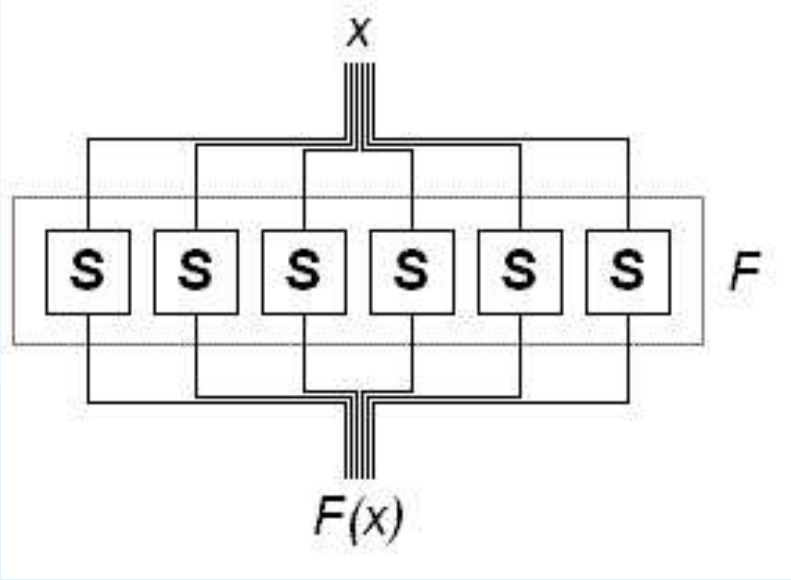


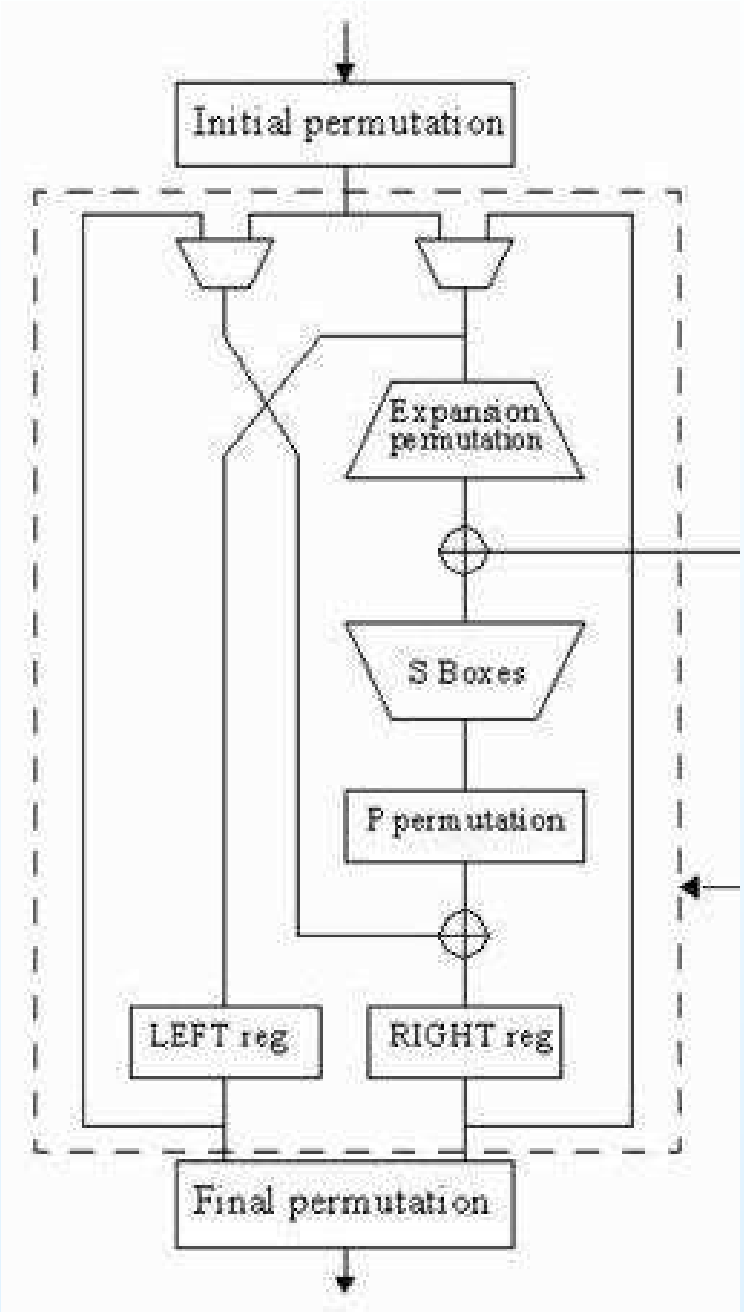

- $F_E(x,k) = R_r(R_{r-1}(...(R_1(x,k_1)...),k_{r-1}),k_r)$

# Round Structure

- Several linear “diffusion" steps
- Key addition (also linear)
- Nonlinear substitution “confusion" step (S-box)

$$R(x) = L_2(F(L_1(x) + k))$$



## Building a Model

- We want to cleanly represent the component functions
- All act on fixed-length binary strings
- Dominant operation: exclusive or (xor), aka bitwise addition without carry

## Building a Model

- We want to cleanly represent the component functions
- All act on fixed-length binary strings
- Dominant operation: exclusive or (xor), aka bitwise addition without carry

- **Option**: Vector space over $\mathbb{Z}_2$ has the correct $+$ operator

## Building a Model

- We want to cleanly represent the component functions
- All act on fixed-length binary strings
- Dominant operation: exclusive or (xor), aka bitwise addition without carry

- **Option**: Vector space over $\mathbb{Z}_2$ has the correct $+$ operator
- **Option**: The group $\mathbb{Z}_2 \oplus \mathbb{Z}_2 \oplus ... \oplus \mathbb{Z}_2$, pretty much the same thing as above

## Building a Model

- We want to cleanly represent the component functions
- All act on fixed-length binary strings
- Dominant operation: exclusive or (xor), aka bitwise addition without carry

- **Option**: Vector space over $\mathbb{Z}_2$ has the correct $+$ operator
- **Option**: The group $\mathbb{Z}_2 \oplus \mathbb{Z}_2 \oplus ... \oplus \mathbb{Z}_2$, pretty much the same thing as above
- **Option**: The field GF$(2^n)$. Behaves exactly like the two previous options under the $+$ operation, but also has the added machinery of a $\times$ operator, which gives us access to nonlinearity.

# Building a Model

- We want to cleanly represent the component functions
- All act on fixed-length binary strings
- Dominant operation: exclusive or (xor), aka bitwise addition without carry

- **Option**: Vector space over $\mathbb{Z}_2$ has the correct $+$ operator
- **Option**: The group $\mathbb{Z}_2 \oplus \mathbb{Z}_2 \oplus ... \oplus \mathbb{Z}_2$, pretty much the same thing as above
- **Option**: The field GF$(2^n)$. Behaves exactly like the two previous options under the $+$ operation, but also has the added machinery of a $\times$ operator, which gives us access to nonlinearity.

- **Winner: The Field!** (Fields are awesome.)

## Multiplication on GF$(2^n)$

- So what IS $\times$ on GF$(2^n)$?
- Polynomial multiplication! (clearly...)
- Change representations from $n$-dimensional vectors to degree-$n-1$ polynomials:

$$(a_0, a_1, ..., a_{n-1}) \rightarrow a_0 + a_1 x + a_2 x^2 + ... + a_{n-1} x^{n-1}$$

- The multiplication operation is defined by an irreducible polynomial $p(x)$ of degree $n$. $a \times b = ab \pmod{p}$.
- This is a Big Deal. It gives us a wealth of additional functions applied to what were previously just vectors.
- Of particular interest to us are power functions applied to these polynomials.
- Note that on GF$(2^n)$, squaring is a linear operation. ($(a+b)^2 = a^2 + b^2$) (Tell me **that's** not fun to say.)

## Functions on GF$(2^n)$

- In general: $F(x)$ : GF$(2^n) \rightarrow$ GF$(2^m)$
- Usually $m = n$ or $m = 1$
- When $m = n$, usually invertible (a permutation of GF$(2^n)$)
- When $m = 1$, $F(x)$ is called a **boolean function**, and is usually denoted with a lower-case function name.
- The general case can be described in terms of Boolean functions:

$$F(x) = (f_1(x), f_2(x), f_3(x), ..., f_m(x))$$

(Each boolean function provides a single 'bit' of the total output.)

# Boolean Functions

$$f(x) : \mathrm{GF}(2^n) \rightarrow \mathrm{GF}(2)$$

- **Linear boolean functions**: If $\exists t \in \mathrm{GF}(2^n)$ such that $f(x) = t_0 x_0 + ... + t_{n-1} x_{n-1} = t^{\mathsf{T}} x$.
- **Affine boolean functions**: If $\exists t \in \mathrm{GF}(2^n), b \in \mathrm{GF}(2)$ such that $f(x) = t_0 x_0 + ... + t_{n-1} x_{n-1} + b = t^{\mathsf{T}} x + b$.
- There are thus $2^n$ linear boolean functions, $2^{n+1}$ affine boolean functions, and $2^{2^n}$ boolean functions on $\mathrm{GF}(2^n)$.
- Referred to by $\mathcal{L}_n$, $\mathcal{A}_n$, and $\mathcal{B}_n$, respectively.

## Distance on $\mathcal{B}_n$

$$d(f,g) = \#\{x \in \mathsf{GF}(2^n) : f(x) \neq g(x)\}$$

This is called the **hamming distance**, and may be recognizable as the taxicab metric.

## Distance on $\mathcal{B}_n$

$$d(f,g) = \#\{x \in \mathsf{GF}(2^n) : f(x) \neq g(x)\}$$

This is called the **hamming distance**, and may be recognizable as the taxicab metric.

$$d(f,G) = \min\{d(f,g) : g \in G\}.$$

gives us the standard distance-to-sets capability.

## Distance on $\mathcal{B}_n$

$$d(f,g) = \#\{x \in \mathsf{GF}(2^n) : f(x) \neq g(x)\}$$

This is called the **hamming distance**, and may be recognizable as the taxicab metric.

$$d(f,G) = \min\{d(f,g) : g \in G\}.$$

gives us the standard distance-to-sets capability. We can then use this to define the nonlinearity of boolean functions:

$$\mathcal{N}(f) = d(f, \mathcal{A}_n)$$

Eventually we'll want a nonlinearity definition for non-boolean functions as well, but we need more information in order to choose a good one.

## (Carpenter's?) Toolbox

We need some stuff in the way of tool functions before we can proceed to the juicy stuff. We'll start with the trace:

**Definition.** The (field) **trace** of $x \in \mathrm{GF}(2^n)$ is a mapping from $\mathrm{GF}(2^n)$ into itself, given by

$$\mathrm{Tr}(x) = \sum_{i=0}^{n-1} x^{2^i}.$$

- For all $x \in \mathrm{GF}(2^n)$, $\mathrm{Tr}(x) \in \{0, 1\}$.
- For all $x, y \in \mathrm{GF}(2^n)$, $\mathrm{Tr}(x + y) = \mathrm{Tr}(x) + \mathrm{Tr}(y)$.
- For all $x \in \mathrm{GF}(2^n)$, $\mathrm{Tr}(x^2) = (\mathrm{Tr}(x))^2 = \mathrm{Tr}(x)$.

The set of functions $\{x \mapsto \mathrm{Tr}(\omega x) \ : \ \omega \in \mathrm{GF}(2^n)\}$ is exactly $\mathcal{L}_n$.

# The Toolbox II: Revenge of the Walsh Transform

**Definition.** The **Walsh transform** $\mathsf{w}(f)$ of $f \in \mathcal{B}_n$ is defined to be:

$$(\mathsf{w}(f))(t) = \hat{\mathsf{F}}(t) = \sum_{x \in \mathsf{GF}(2^n)} (-1)^{f(x)} (-1)^{t^{\mathsf{T}} x}, \quad t \in \mathsf{GF}(2^n).$$

The Walsh transform is essentially the Fourier transform on $\mathsf{GF}(2^n)$. (Note that for integer inputs, all that $e^{\pi i x}$ stuff becomes $(-1)^x$.)

Most of the standard Fourier stuff still plays nice, including the Convolution Theorem and Parseval's Theorem, which get used in the bowels of a few proofs, but I'll spare you for the time being.

# The Walsh Transform and Nonlinearity

## Why do we care about the Walsh transform?

# The Walsh Transform and Nonlinearity

Why do we care about the Walsh transform? I'm glad you asked!

# The Walsh Transform and Nonlinearity

Why do we care about the Walsh transform? I'm glad you asked! Let $\hat{\mathsf{F}}$ be the Walsh transform of $f$. Then

$$\hat{\mathsf{F}}(t) = \sum_{x \in \mathsf{GF}(2^n)} (-1)^{f(x)} (-1)^{t^{\mathsf{T}} a} = \sum_{\substack{x \in \mathsf{GF}(2^n) \\ f(x) = t^{\mathsf{T}} x}} (1) + \sum_{\substack{x \in \mathsf{GF}(2^n) \\ f(x) \neq t^{\mathsf{T}} x}} (-1)$$

The Walsh transform is directly related to the distance of a function from the linear functions. By some minor manipulation, we get that the distance of a boolean function $f$ from the set of affine boolean functions is

$$d(f, \mathcal{A}_n) = 2^{n-1} - \frac{1}{2} \max_{t \in \mathsf{GF}(2^n)} |\hat{\mathsf{F}}(t)|$$

This is an alternate way to calculate the nonlinearity of $f$!

## What are we doing again?

We are after a secure S-box function. To get this, we need to:

- Determine what being secure means
- Generate criteria for a viable function
- Find a function that has those criteria in spades

No sweat, right?

We'll begin by studying the major attacks against S-box functions.

# Cryptanalysis


"The best system is to use a simple, well understood algorithm which relies on the security of a key rather than the algorithm itself. This means if anybody steals a key, you could just roll another and they have to **start all over**."

– Andrew Carol

# Cryptanalysis


All the security in a cipher should be in the secrecy of the key. It is bad practice to rely on, for instance, the secrecy of the algorithm used, or the particular settings, or even the secrecy of the plaintext.

As such, we cede the attacker full knowledge of the cipher, as well as the ability to input values into it and see what outputs are generated. This is called a **known plaintext attack**.

The goal of the attacker, then, is to utilize a large pool of (input, output) pairs to recover the key (or all the round keys). To be an effective attack, the attacker should use *fewer* encryptions than a brute-force search of the keyspace would require.

## Linear Cryptanalysis

- The first of two attacks on the S-box function $\mathcal{S}(x)$ we'll consider.
- Goal: $a \in \mathrm{GF}(2^n), b \in \mathrm{GF}(2^m)$ such that

$$a^{\mathsf{T}} x = b^{\mathsf{T}} \mathcal{S}(x) \pmod 2$$

holds with probability $\frac{1}{2} + \epsilon$

- The larger in magnitude $\epsilon$ is, the better the linear approximation.
- Extend from $\mathcal{S}(x)$ to to the whole nonlinear function $F(x)$, and then to the round function $R(x)$, including the key addition.
- End result:

$$d^{\mathsf{T}} k = a^{\mathsf{T}} x + b^{\mathsf{T}} R(x) \pmod 2$$

## Differential Cryptanalysis

- Now we look for correlations between input and output **differences**
- Given inputs $x_1, x_2 \in \mathrm{GF}(2^n)$, and outputs $y_1 = \mathcal{S}(x_1), y_2 = \mathcal{S}(x_2) \in \mathrm{GF}(2^m)$, calculate

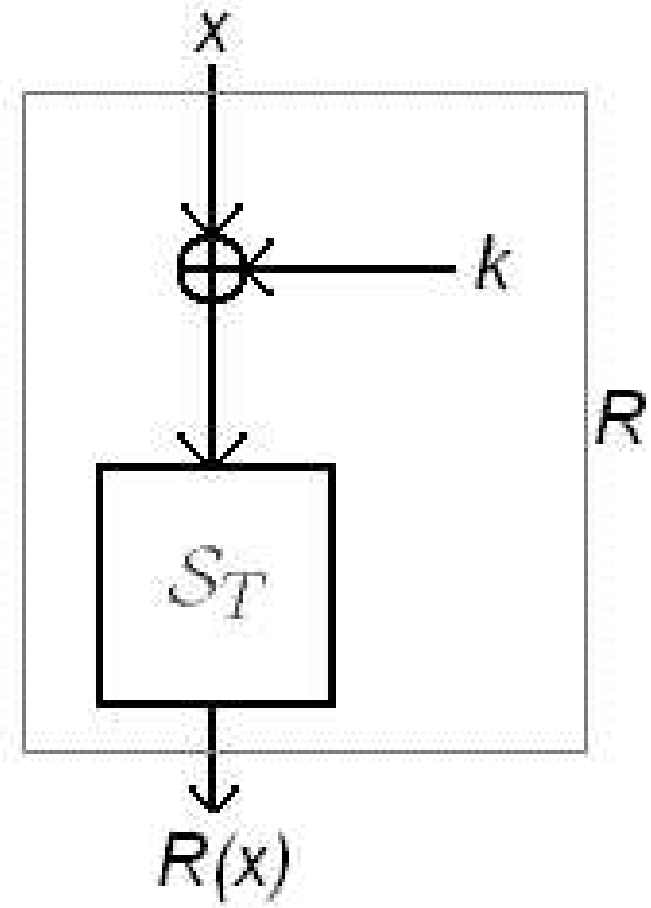
$$x' = x_1 + x_2 \qquad y' = y_1 + y_2$$

- Given a **round** input and output difference, push inward so that you're as close to the key and nonlinear function as possible.
- Only some input differences will be possible, given a known output difference. Each will suggest a possible key.
- repeat the test with varying input/output differences. The real key will be the one suggested by **all** tests.

# Bob

- Toy round function.
- $R(x) : \mathrm{GF}(2^5) \rightarrow \mathrm{GF}(2^5)$
- $R(x) = \mathcal{S}_T(x + k)$
- No linear diffusion functions, since we're not really interested in them.
- Put on your black hats, we're about to break this thing.

- (Don't tell anyone, but the key we're trying to recover is actually 10101)



# Definition of $\mathcal{S}_T$

| Definition of $\mathcal{S}_T$ (lookup table): | | |
|---|---|---|
| $\mathcal{S}_T(00000) = 01001$ | $\mathcal{S}_T(00001) = 01100$ | $\mathcal{S}_T(00010) = 10101$ |
| $\mathcal{S}_T(00011) = 01101$ | $\mathcal{S}_T(00100) = 00110$ | $\mathcal{S}_T(00101) = 10100$ |
| $\mathcal{S}_T(00110) = 00010$ | $\mathcal{S}_T(00111) = 11001$ | $\mathcal{S}_T(01000) = 10110$ |
| $\mathcal{S}_T(01001) = 01110$ | $\mathcal{S}_T(01010) = 11110$ | $\mathcal{S}_T(01011) = 11010$ |
| $\mathcal{S}_T(01100) = 01011$ | $\mathcal{S}_T(01101) = 11100$ | $\mathcal{S}_T(01110) = 00011$ |
| $\mathcal{S}_T(01111) = 10111$ | $\mathcal{S}_T(10000) = 00111$ | $\mathcal{S}_T(10001) = 10010$ |
| $\mathcal{S}_T(10010) = 01111$ | $\mathcal{S}_T(10011) = 00001$ | $\mathcal{S}_T(10100) = 11000$ |
| $\mathcal{S}_T(10101) = 00101$ | $\mathcal{S}_T(10110) = 11101$ | $\mathcal{S}_T(10111) = 10000$ |
| $\mathcal{S}_T(11000) = 00100$ | $\mathcal{S}_T(11001) = 10011$ | $\mathcal{S}_T(11010) = 01010$ |
| $\mathcal{S}_T(11011) = 11111$ | $\mathcal{S}_T(11100) = 11011$ | $\mathcal{S}_T(11101) = 00000$ |
| $\mathcal{S}_T(11110) = 01000$ | $\mathcal{S}_T(11111) = 10001$ | |

# Bob vs Linear Cryptanalysis

- Looking for a good $a^{\mathsf{T}}x = b^{\mathsf{T}}\mathcal{S}_T(x)$ approx.
- (Arbitrarily) choose $b = 10000$ (only paying attention to the top-order bit in the output)
- Partial table of information for various $a$:

| | |
|---|---|
| $d(b^{\mathsf{T}}\mathcal{S}_T(x), 10001^{\mathsf{T}}x) = 16$ | $\mathsf{w}(b^{\mathsf{T}}\mathcal{S}_T)(10001) = 0$ |
| $d(b^{\mathsf{T}}\mathcal{S}_T(x), 10010^{\mathsf{T}}x) = 14$ | $\mathsf{w}(b^{\mathsf{T}}\mathcal{S}_T)(10010) = 4$ |
| $d(b^{\mathsf{T}}\mathcal{S}_T(x), 10011^{\mathsf{T}}x) = 14$ | $\mathsf{w}(b^{\mathsf{T}}\mathcal{S}_T)(10011) = 4$ |
| $d(b^{\mathsf{T}}\mathcal{S}_T(x), 10100^{\mathsf{T}}x) = 18$ | $\mathsf{w}(b^{\mathsf{T}}\mathcal{S}_T)(10100) = -4$ |
| $d(b^{\mathsf{T}}\mathcal{S}_T(x), 10101^{\mathsf{T}}x) = 26$ | $\mathsf{w}(b^{\mathsf{T}}\mathcal{S}_T)(10101) = -20$ |
| $d(b^{\mathsf{T}}\mathcal{S}_T(x), 10110^{\mathsf{T}}x) = 12$ | $\mathsf{w}(b^{\mathsf{T}}\mathcal{S}_T)(10110) = 8$ |

# Bob vs Linear Cryptanalysis

- Looking for a good $a^{\mathsf{T}}x = b^{\mathsf{T}}\mathcal{S}_T(x)$ approx.
- (Arbitrarily) choose $b = 10000$ (only paying attention to the top-order bit in the output)
- Partial table of information for various $a$:

| | |
|---|---|
| $d(b^{\mathsf{T}}\mathcal{S}_T(x), 10001^{\mathsf{T}}x) = 16$ | $\mathsf{w}(b^{\mathsf{T}}\mathcal{S}_T)(10001) = 0$ |
| $d(b^{\mathsf{T}}\mathcal{S}_T(x), 10010^{\mathsf{T}}x) = 14$ | $\mathsf{w}(b^{\mathsf{T}}\mathcal{S}_T)(10010) = 4$ |
| $d(b^{\mathsf{T}}\mathcal{S}_T(x), 10011^{\mathsf{T}}x) = 14$ | $\mathsf{w}(b^{\mathsf{T}}\mathcal{S}_T)(10011) = 4$ |
| $d(b^{\mathsf{T}}\mathcal{S}_T(x), 10100^{\mathsf{T}}x) = 18$ | $\mathsf{w}(b^{\mathsf{T}}\mathcal{S}_T)(10100) = -4$ |
| $d(b^{\mathsf{T}}\mathcal{S}_T(x), 10101^{\mathsf{T}}x) = 26$ | $\mathsf{w}(b^{\mathsf{T}}\mathcal{S}_T)(10101) = -20$ |
| $d(b^{\mathsf{T}}\mathcal{S}_T(x), 10110^{\mathsf{T}}x) = 12$ | $\mathsf{w}(b^{\mathsf{T}}\mathcal{S}_T)(10110) = 8$ |

# Bob vs Linear Cryptanalysis

- Looking for a good $a^{\mathsf{T}}x = b^{\mathsf{T}}\mathcal{S}_T(x)$ approx.
- (Arbitrarily) choose $b = 10000$ (only paying attention to the top-order bit in the output)
- Partial table of information for various $a$:

| | |
|---|---|
| $d(b^{\mathsf{T}}\mathcal{S}_T(x), 10001^{\mathsf{T}}x) = 16$ | $\mathsf{w}(b^{\mathsf{T}}\mathcal{S}_T)(10001) = 0$ |
| $d(b^{\mathsf{T}}\mathcal{S}_T(x), 10010^{\mathsf{T}}x) = 14$ | $\mathsf{w}(b^{\mathsf{T}}\mathcal{S}_T)(10010) = 4$ |
| $d(b^{\mathsf{T}}\mathcal{S}_T(x), 10011^{\mathsf{T}}x) = 14$ | $\mathsf{w}(b^{\mathsf{T}}\mathcal{S}_T)(10011) = 4$ |
| $d(b^{\mathsf{T}}\mathcal{S}_T(x), 10100^{\mathsf{T}}x) = 18$ | $\mathsf{w}(b^{\mathsf{T}}\mathcal{S}_T)(10100) = -4$ |
| $d(b^{\mathsf{T}}\mathcal{S}_T(x), 10101^{\mathsf{T}}x) = 26$ | $\mathsf{w}(b^{\mathsf{T}}\mathcal{S}_T)(10101) = -20$ |
| $d(b^{\mathsf{T}}\mathcal{S}_T(x), 10110^{\mathsf{T}}x) = 12$ | $\mathsf{w}(b^{\mathsf{T}}\mathcal{S}_T)(10110) = 8$ |

So $d(10000^{\mathsf{T}}\mathcal{S}_T(x), 10101^{\mathsf{T}}x + 1) = 32 - 26 = 6$

Thus $10000^{\mathsf{T}}\mathcal{S}_T(x) = 10101^{\mathsf{T}}x + 1$ with probability $81.25\%$

# Bob vs Linear Cryptanalysis II

So, we have an approximation for $\mathcal{S}_T(x)$:

$$10000^{\mathsf{T}} \mathcal{S}_T(x) = 10101^{\mathsf{T}} x + 1$$

## Bob vs Linear Cryptanalysis II

So, we have an approximation for $\mathcal{S}_T(x)$:

$$10000^{\mathsf{T}}\mathcal{S}_T(x) = 10101^{\mathsf{T}}x + 1$$

But this can be extended to an approximation for the full round, since the input to the S-box is really $x + k$, where $x$ is the round input. The approximation then becomes:

$$10101^{\mathsf{T}}k = 10000^{\mathsf{T}}R(x) + 10101^{\mathsf{T}}x + 1$$

Which still holds with probability $81.25\%$

So, let's throw some inputs at this and see if a trend emerges.

# Bob vs Linear Cryptanalysis III

Affine approximation: $10101^{\mathsf{T}}k = 10000^{\mathsf{T}}R(x) + 10101^{\mathsf{T}}x + 1$

| $x$ | $R(x)$ | $10101^{\mathsf{T}}x$ | $10000^{\mathsf{T}}R(x)$ |
|---|---|---|---|
| 01001 | 11011 | 1 | 1 |
| 00001 | 11000 | 1 | 1 |
| 00000 | 00101 | 0 | 0 |
| 00010 | 10000 | 0 | 1 |
| 11111 | 11110 | 1 | 1 |

# Bob vs Linear Cryptanalysis III

Affine approximation: $10101^{\mathsf{T}}k = 10000^{\mathsf{T}}R(x) + 10101^{\mathsf{T}}x + 1$

| $x$ | $R(x)$ | $10101^{\mathsf{T}}x$ | $10000^{\mathsf{T}}R(x)$ |
|---|---|---|---|
| 01001 | 11011 | 1 | 1 |
| 00001 | 11000 | 1 | 1 |
| 00000 | 00101 | 0 | 0 |
| 00010 | 10000 | 0 | 1 |
| 11111 | 11110 | 1 | 1 |

These five $x$ values suggest $10101^{\mathsf{T}}k = 1$ 80% of the time, so it is likely that the parity of the first, third, and fifth bits of $k$ is odd. If we had constructed other approximations, we could have recovered more information from the same data and recovered the full key (5 key bits would require 5 ind. approximations).

## Why Bob Broke

Bob fell to our attack because we were able to construct linear approximations of his nonlinear component. The form this approximations took tells us what kind of nonlinear definition we need.

- A “linear combination of input/output bits" is a boolean function, so the nonlinearity of functions in general should be definable in terms of nonlinearity of boolean functions.
- Specifically, we want no linear combination of the output bits of $\mathcal{S}$ to be near to an affine boolean function. So!

## Why Bob Broke

Bob fell to our attack because we were able to construct linear approximations of his nonlinear component. The form this approximations took tells us what kind of nonlinear definition we need.

- A “linear combination of input/output bits" is a boolean function, so the nonlinearity of functions in general should be definable in terms of nonlinearity of boolean functions.
- Specifically, we want no linear combination of the output bits of $\mathcal{S}$ to be near to an affine boolean function. So!

**Definition.** For function $F : \mathrm{GF}(2^n) \to \mathrm{GF}(2^m)$ the **nonlinearity** of $F$ is:

$$\mathcal{N}(F) = \min_{\substack{b \in \mathrm{GF}(2^m) \\ b \neq 0}} \mathcal{N}(b^{\mathsf{T}} F)$$

Large $\mathcal{N}$ is a requirement for a secure $\mathcal{S}$ function.

## Bob vs Differential Cryptanalysis

**Test case 1:** $x = 11000, x' = 00001$

- $R(x) = 11100, R(x + x') = 01011 \rightarrow y' = 10111$
- Pairs xor table says four possible values for x+k:
  - 01100, 01101, 11000, 11001
- Each suggests a possible key value:
  - 10100, 10101, 00000, 00001

# Bob vs Differential Cryptanalysis

**Test case 1:** $x = 11000, x' = 00001$

- $R(x) = 11100, R(x + x') = 01011 \rightarrow y' = 10111$
- Pairs xor table says four possible values for x+k:
  - 01100, 01101, 11000, 11001
- Each suggests a possible key value:
  - 10100, 10101, 00000, 00001

| Possible key values: |
|---|
| 10100 |
| 10101 |
| 00000 |
| 00001 |

# Bob vs Differential Cryptanalysis

**Test case 2:** $x = 01001, x' = 01110$

- $R(x) = 11011, R(x + x') = 01111 \rightarrow y' = 10100$
- Pairs xor table says four possible values for x+k:
  - 00110, 01000, 10010, 11100
- Each suggests a possible key value:
  - ~~01111~~, 00001, ~~11011~~, 10101

| Possible key values: |
|---|
| 10100 |
| 10101 |
| 00000 |
| 00001 |

# Bob vs Differential Cryptanalysis

**Test case 2:** $x = 01001, x' = 01110$

- $R(x) = 11011, R(x + x') = 01111 \rightarrow y' = 10100$
- Pairs xor table says four possible values for x+k:
  - 00110, 01000, 10010, 11100
- Each suggests a possible key value:
  - ~~01111~~, 00001, ~~11011~~, 10101

| Possible key values: |
| --- |
| ~~10100~~ |
| 10101 |
| ~~00000~~ |
| 00001 |

# Bob vs Differential Cryptanalysis

**Test case 3:** $x = 01100, x' = 01010$

- $R(x) = 10011, R(x + x') = 00001 \rightarrow y' = 10010$
- Pairs xor table says two possible values for x+k:
  - 10011, 11001
- Each suggests a possible key value:
  - ~~11111~~, 10101

| Possible key values: |
| --- |
| ~~10100~~ |
| 10101 |
| ~~00000~~ |
| 00001 |

# Bob vs Differential Cryptanalysis

**Test case 3:** $x = 01100, x' = 01010$

- $R(x) = 10011, R(x + x') = 00001 \rightarrow y' = 10010$
- Pairs xor table says two possible values for x+k:
  - 10011, 11001
- Each suggests a possible key value:
  - ~~11111~~, 10101

| Possible key values: |
| --- |
| ~~10100~~ |
| 10101 |
| ~~00000~~ |
| ~~00001~~ |

# Bob vs Differential Cryptanalysis

**Test case 3:** $x = 01100, x' = 01010$

- $R(x) = 10011, R(x + x') = 00001 \rightarrow y' = 10010$
- Pairs xor table says two possible values for x+k:
  - 10011, 11001
- Each suggests a possible key value:
  - ~~11111~~, 10101

Key recovered! k = 10101

| Possible key values: |
| --- |
| ~~10100~~ |
| 10101 |
| ~~00000~~ |
| ~~00001~~ |

## Poor Bob

- Alas, Bob couldn't resist this attack either.
- It is less clear, however, how one **would** resist it.
- In fact, the attack is not resistable on single rounds. Resistance must come from making it difficult to chain the attacks together to leverage against the full cipher.
- The details are obnoxious.
- The only way to prevent this attack from being successful is in keeping correlations between input and output differences as low as possible.
- This makes it difficult to build "differential trails" through the whole cipher, and saves individual rounds from falling to the attack.

## A New Hope

- For Bob to resist this attack, given an input difference $x' \in \mathrm{GF}(2^n)$, there should not be very many $x \in \mathrm{GF}(2^n)$ that lead to the same output difference $y' \in \mathrm{GF}(2^m)$.
- This is formalized in the concept of differential uniformity:

**Definition.** A function $F : \mathrm{GF}(2^n) \to \mathrm{GF}(2^m)$ is called **differentially $\delta$-uniform** if, for all $x' \in \mathrm{GF}(2^n), y' \in \mathrm{GF}(2^m), x' \neq 0$,

$$\#\{x \in \mathrm{GF}(2^n) : F(x + x') + F(x) = y'\} \leq \delta$$

Small $\delta$ is a requirement for a secure $\mathcal{S}$ function.

## Moving Right Along...

So, Bob is doomed. His randomly-generated $\mathcal{S}_T$ function was not suitable. But we've learned from his mistakes. Our next S-box will be based off of a function that satisfies the following conditions:

**Goal A.** A suitable S-box function $\mathcal{S}$: GF$(2^n) \rightarrow$ GF$(2^m)$ should have the property that $\mathcal{N}(\mathcal{S})$ is maximal or near-maximal. Specifically, $\mathcal{N}(\mathcal{S})$ should be close to the upper bound $2^{n-1} - 2^{\frac{n}{2}-1}$.

**Goal B.** A suitable S-box function $\mathcal{S}$: GF$(2^n) \rightarrow$ GF$(2^m)$ should be differentially $\delta$-uniform for as small a $\delta$ as possible.

**Now we're getting somewhere!**

## And the Winner is...

There are four big hurking proofs behind the following, but the margin is... you know.

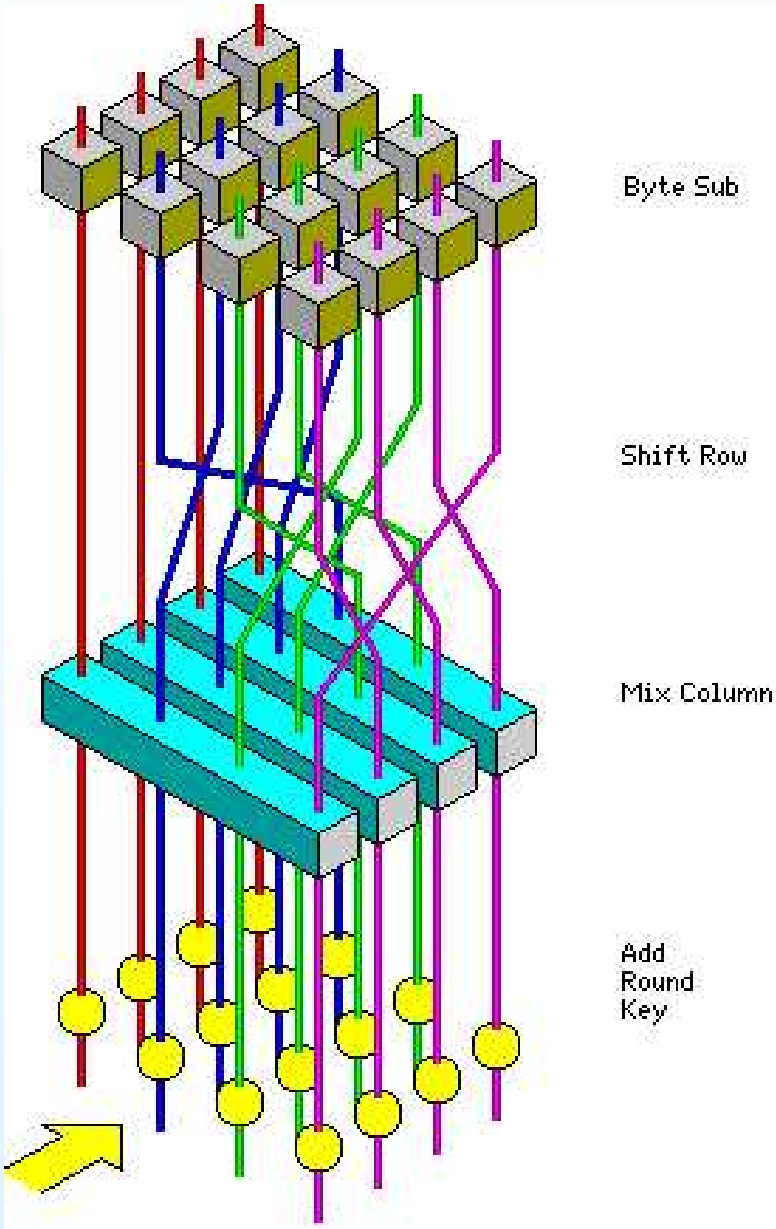
**Proposition.** *Let $F(x) = x^{2^k+1}$ for $x \in GF(2^n)$. Let $s = \gcd(k, n)$. If $F$ is a permutation, then $\mathcal{N}(F) = 2^{n-1} - 2^{\frac{n+s}{2}-1}$ and $F$ is differentially $2^s$-uniform.*

**Proposition.** *Let $F(x)$ be the inversion function, defined by $F(x) = x^{-1}$ for $x \neq 0$ and $F(0) = 0$. Then $\mathcal{N}(F) \geq 2^{n-1} - 2^{\frac{n}{2}}$ and $F$ is differentially $4$-uniform.*

The cipher $C^*$ is based on the power functions given above. Rijndael is based on the inversion function.

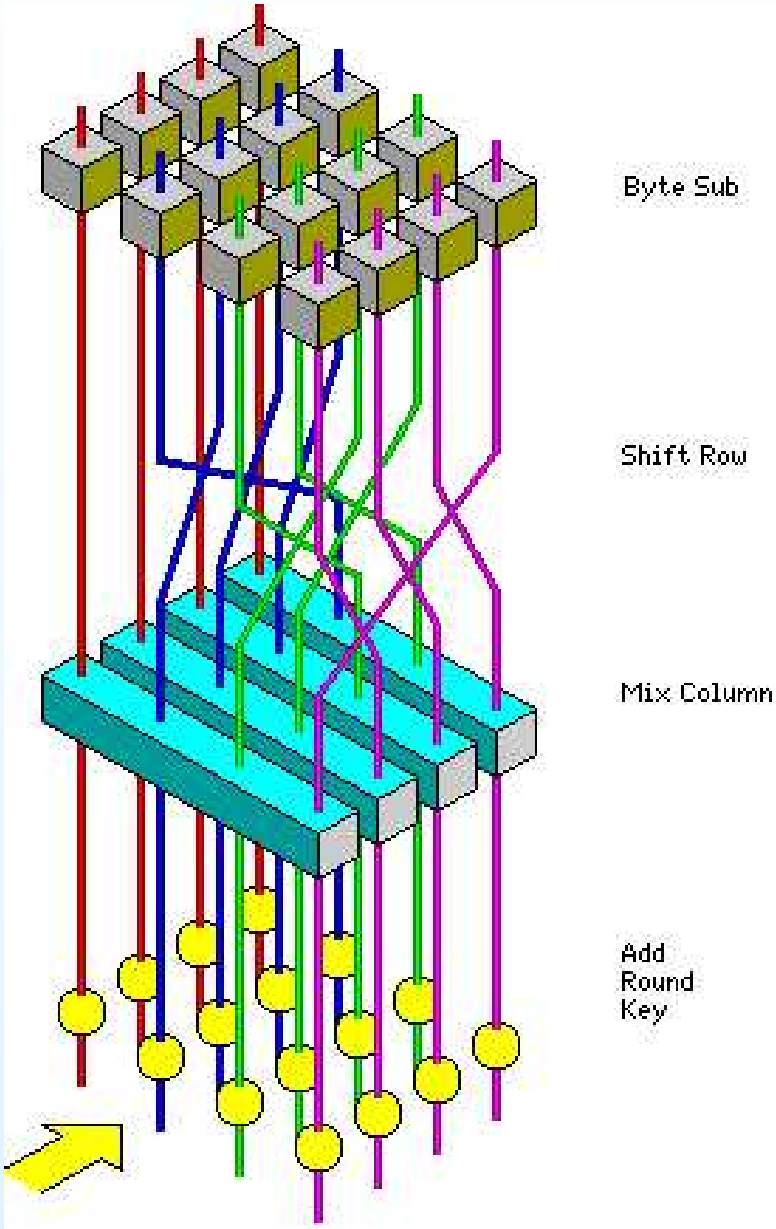
# Did Someone Say Rijndael?

- This is what we've been waiting for



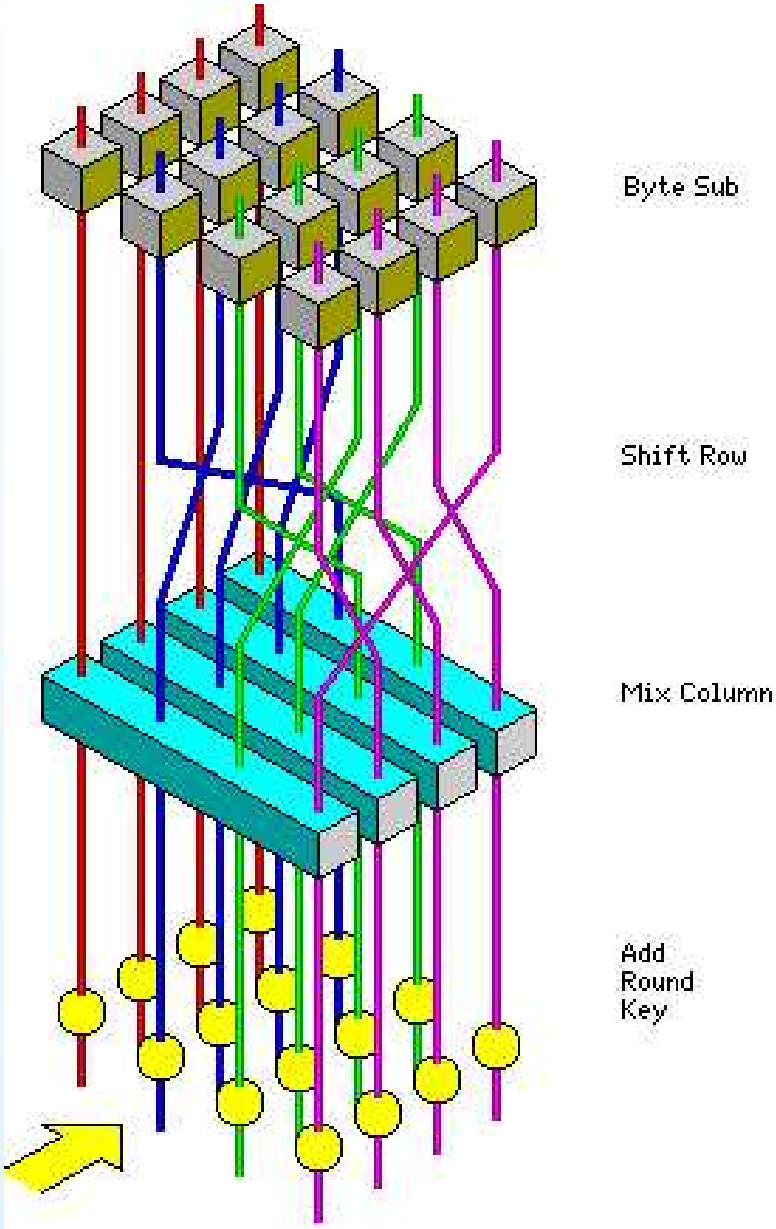
# Did Someone Say Rijndael?

- This is what we've been waiting for
- 1970s: The first public encryption standard: DES



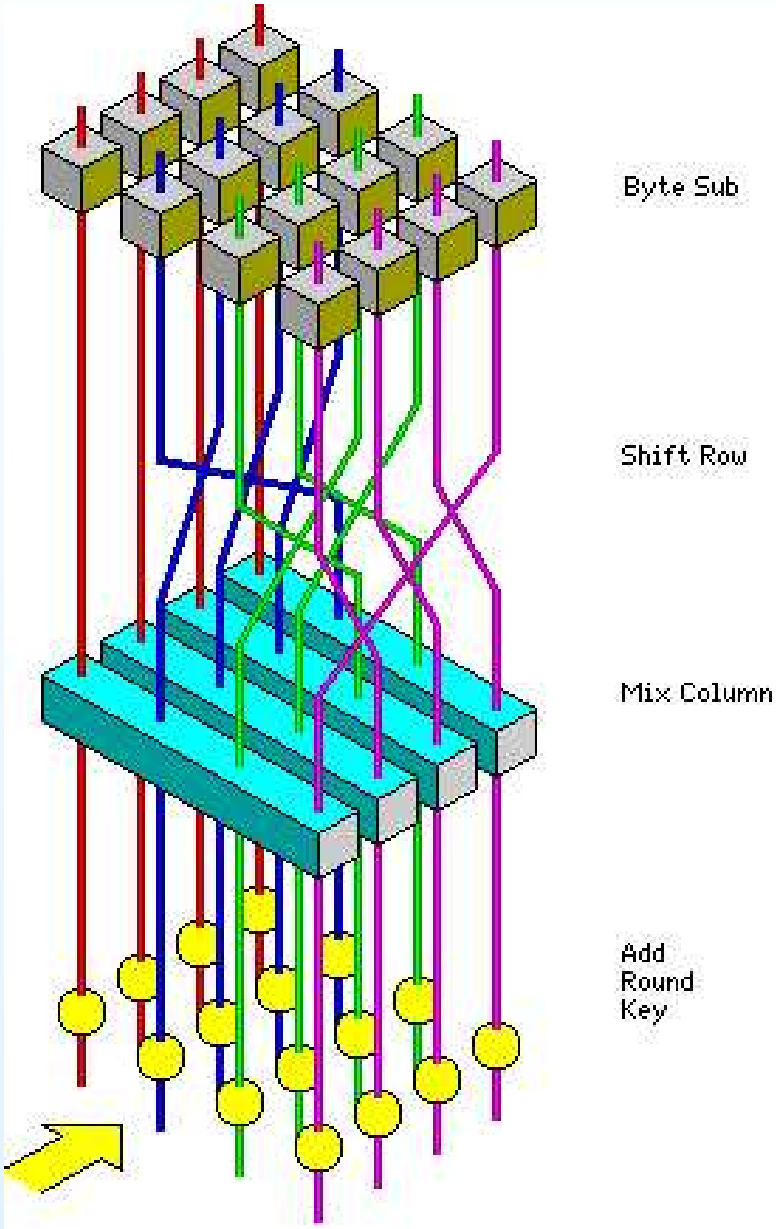
# Did Someone Say Rijndael?

- This is what we've been waiting for
- 1970s: The first public encryption standard: DES
- 1993: DES obsolete- possible to search the full 56-bit keyspace with specialized hardware in hours



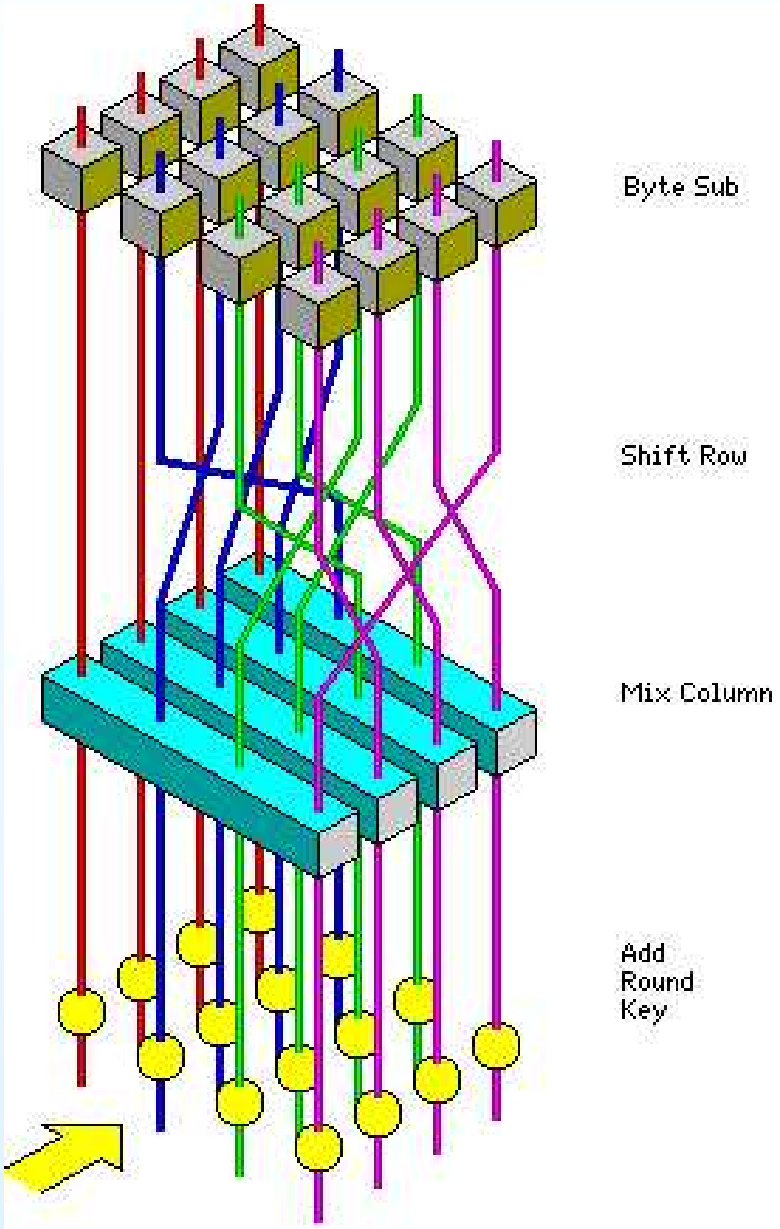
# Did Someone Say Rijndael?

- This is what we've been waiting for
- 1970s: The first public encryption standard: DES
- 1993: DES obsolete- possible to search the full 56-bit keyspace with specialized hardware in hours
- 1997: NIST releases design requirements for a replacement cipher: AES



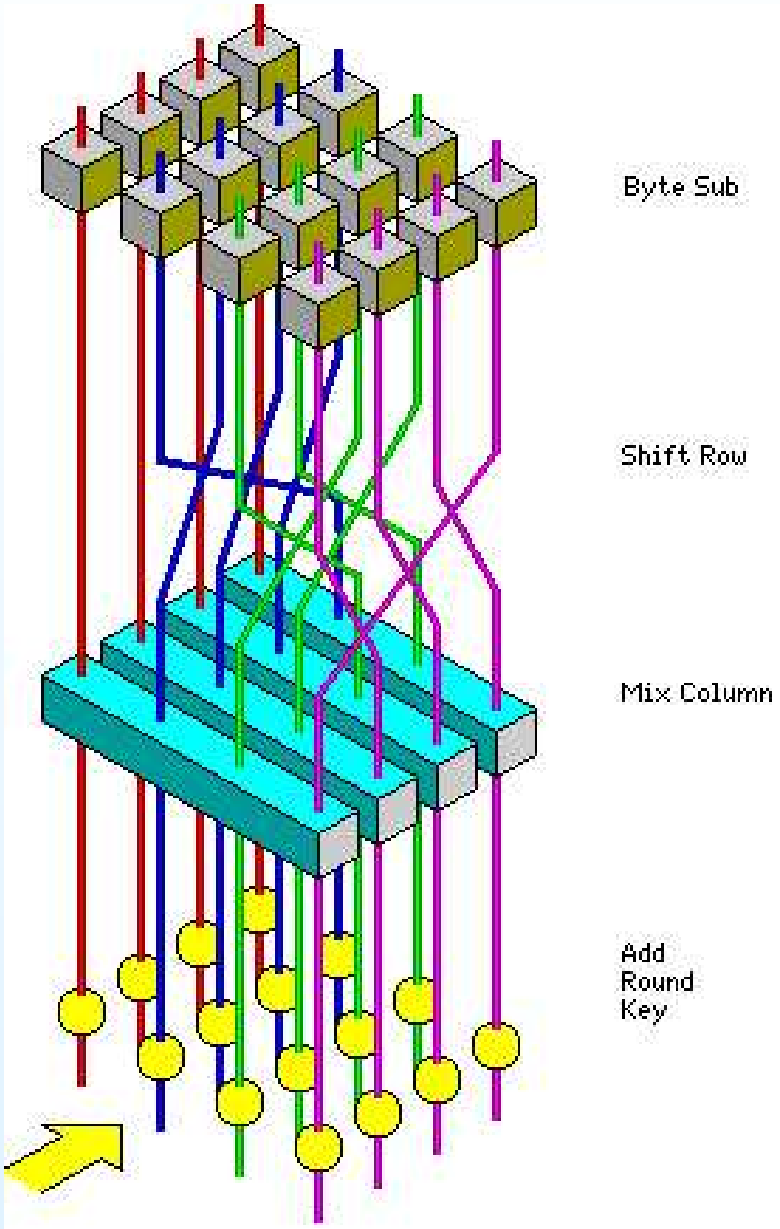
# Did Someone Say Rijndael?

- This is what we've been waiting for
- 1970s: The first public encryption standard: DES
- 1993: DES obsolete- possible to search the full 56-bit keyspace with specialized hardware in hours
- 1997: NIST releases design requirements for a replacement cipher: AES
- Five finalists, all highly secure, state of the art: MARS, RC6, Rijndael, Serpent, Twofish



# Did Someone Say Rijndael?

- This is what we've been waiting for
- 1970s: The first public encryption standard: DES
- 1993: DES obsolete- possible to search the full 56-bit keyspace with specialized hardware in hours
- 1997: NIST releases design requirements for a replacement cipher: AES
- Five finalists, all highly secure, state of the art: MARS, RC6, Rijndael, Serpent, Twofish
- 2000: Rijndael wins!

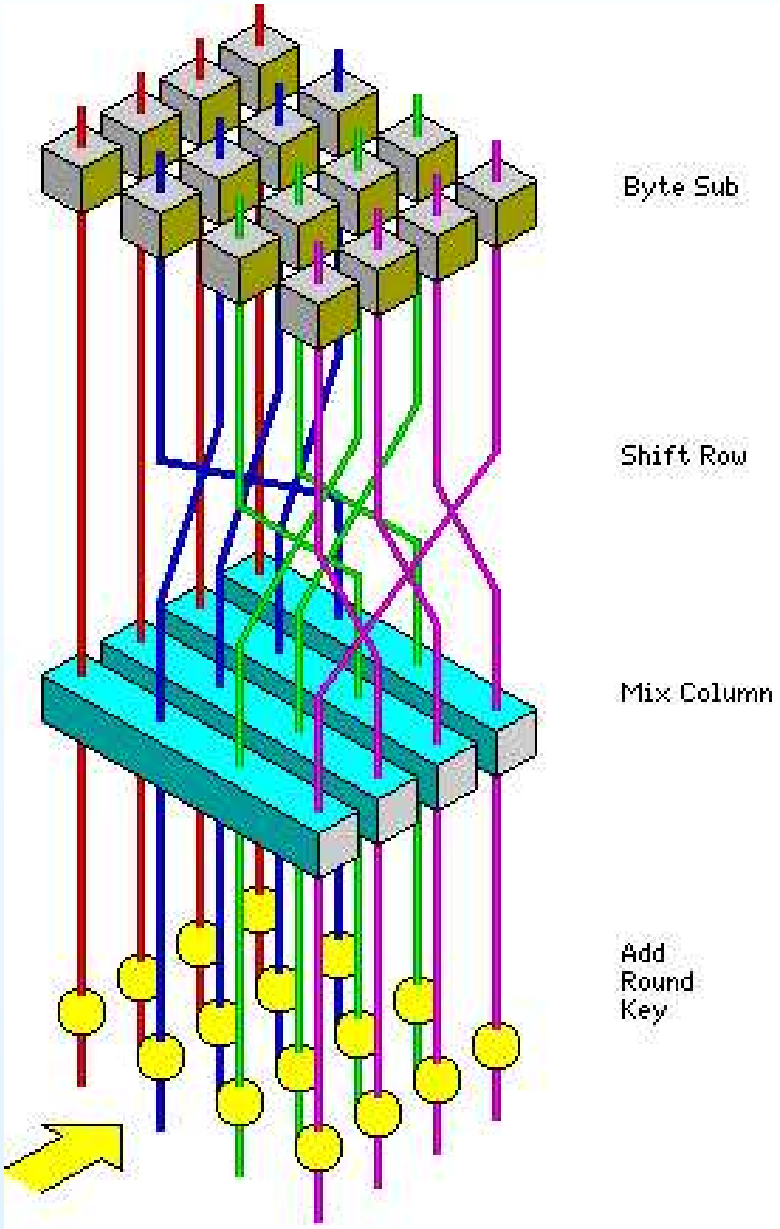


# Pronounced Just Like It's Spelled

$$R(x) = L_2(L_1(F(x))) + k$$

Criteria for Rijndael's $\mathcal{S}$ function:

- Invertibility

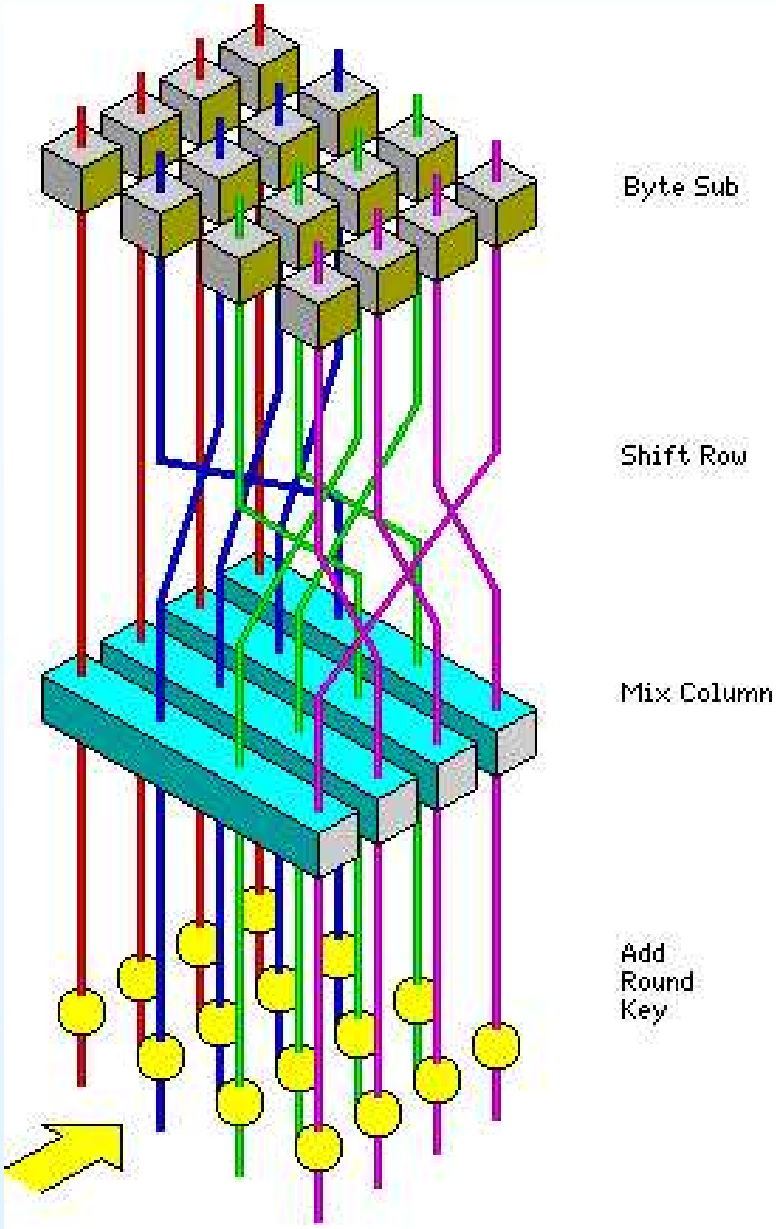


# Pronounced Just Like It's Spelled

$$R(x) = L_2(L_1(F(x))) + k$$

Criteria for Rijndael's $\mathcal{S}$ function:

- Invertibility
- Minimization of the largest non-trivial correlation between linear combinations of input bits and linear combinations of output bits

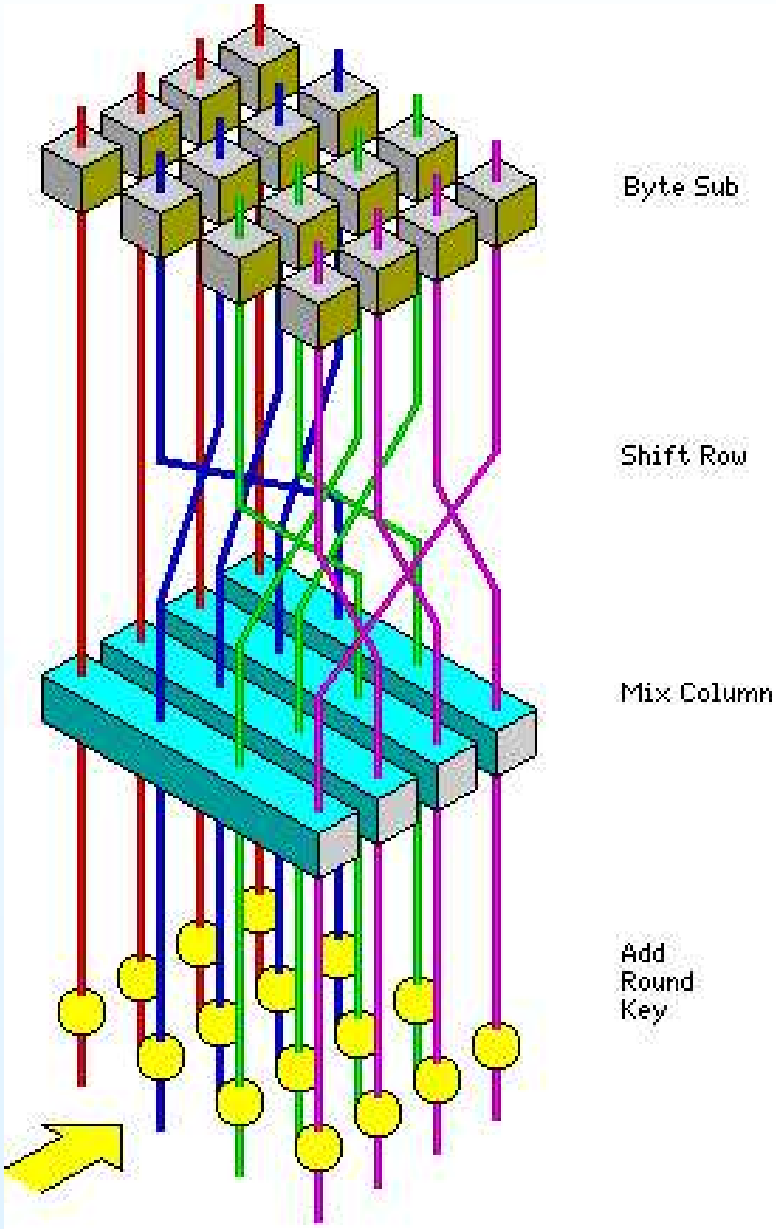


# Pronounced Just Like It's Spelled

$$R(x) = L_2(L_1(F(x))) + k$$

Criteria for Rijndael's $\mathcal{S}$ function:

- Invertibility
- Minimization of the largest non-trivial correlation between linear combinations of input bits and linear combinations of output bits
- Minimization of the largest non-trivial value in the xor table

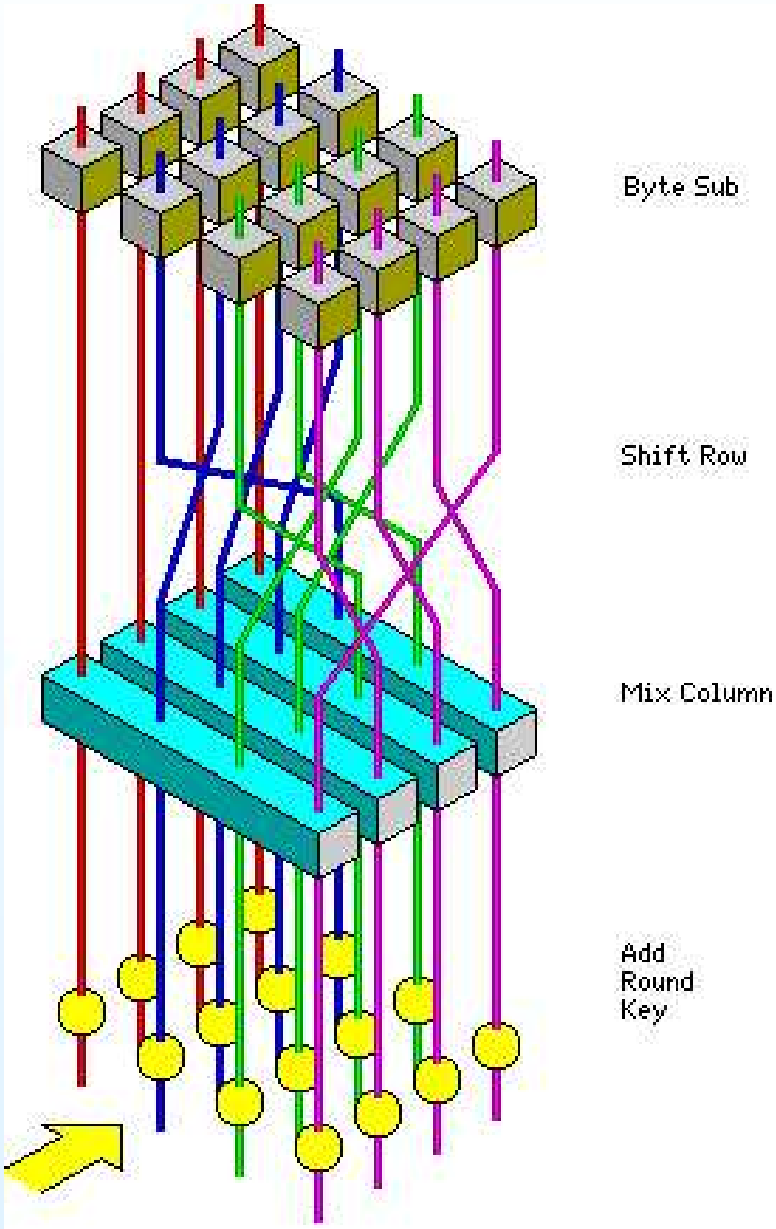


# Pronounced Just Like It's Spelled

$$R(x) = L_2(L_1(F(x))) + k$$

Criteria for Rijndael's $\mathcal{S}$ function:

- Invertibility
- Minimization of the largest non-trivial correlation between linear combinations of input bits and linear combinations of output bits
- Minimization of the largest non-trivial value in the xor table
- Complexity of its algebraic expression in GF$(2^8)$

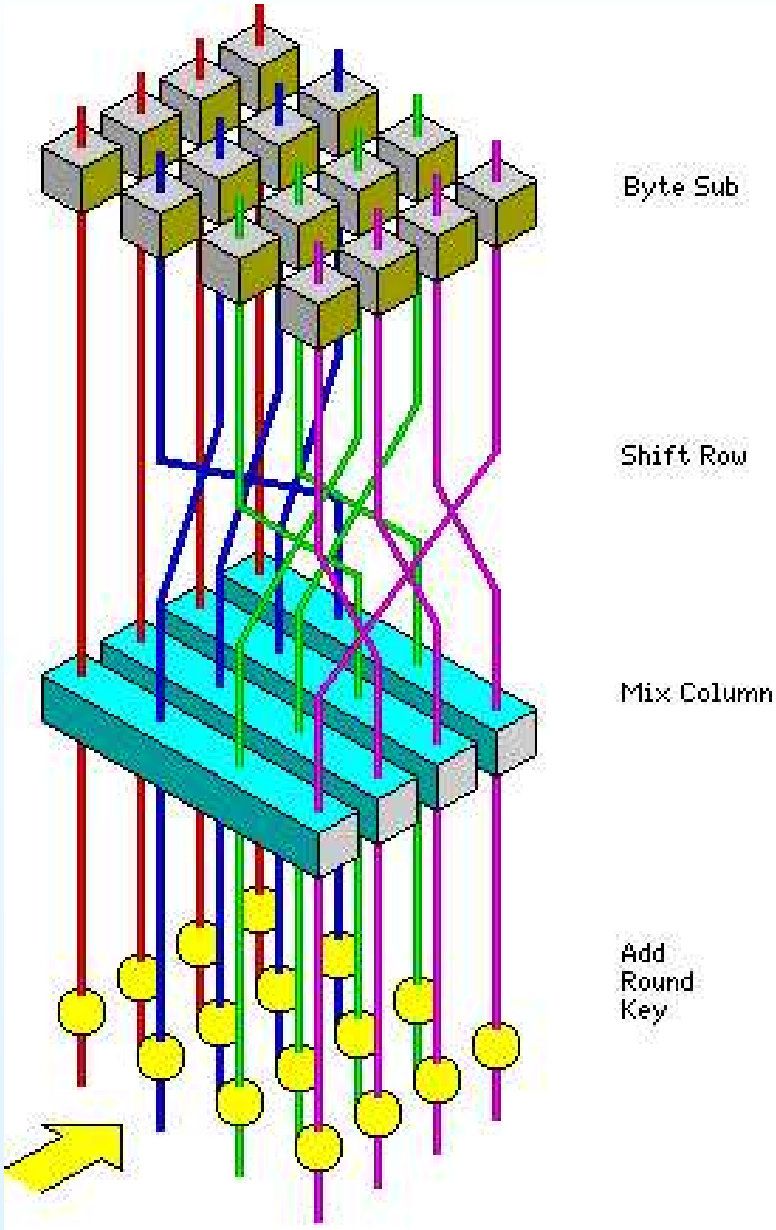


# Pronounced Just Like It's Spelled

$$R(x) = L_2(L_1(F(x))) + k$$

Criteria for Rijndael's $\mathcal{S}$ function:

- Invertibility
- Minimization of the largest non-trivial correlation between linear combinations of input bits and linear combinations of output bits
- Minimization of the largest non-trivial value in the xor table
- Complexity of its algebraic expression in GF$(2^8)$
- Simplicity of description



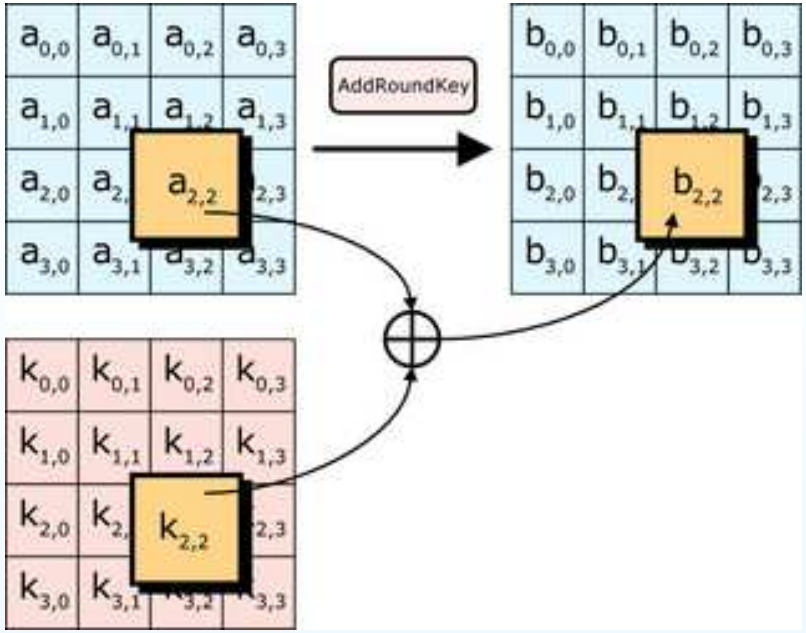
## Math Trumps Black Magic

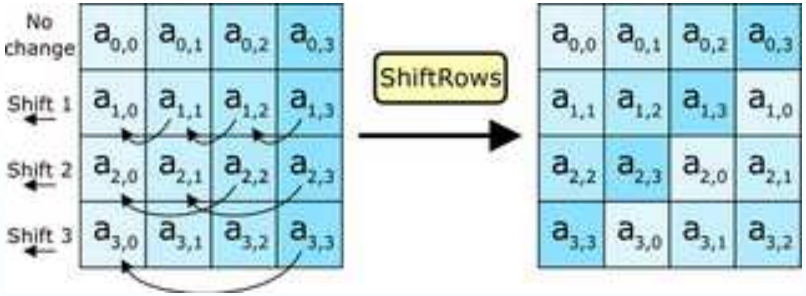
$$\mathcal{S}(x) = S_2(S_1(x)) : \mathsf{GF}(2^8) \mapsto \mathsf{GF}(2^8)$$

- $S_1$ is the inversion function we just described under the modulus $x^8 + x^4 + x^3 + x + 1$
- $S_2$ is an invertible affine mapping $p(x) \mapsto q(x)$

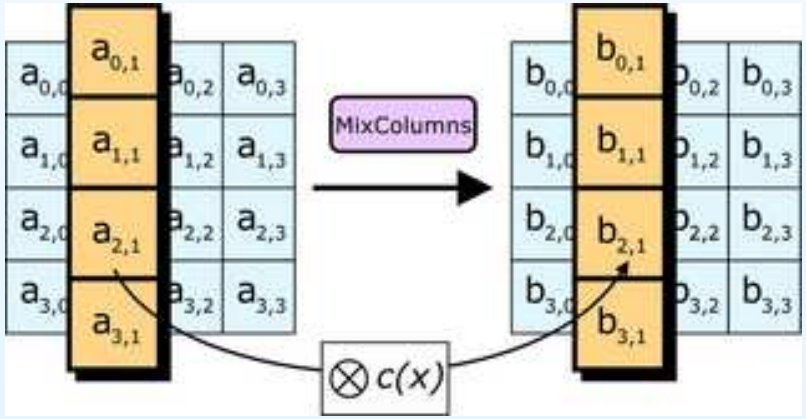
$$q(x) = (x^7 + x^6 + x^2 + x) + p(x)(x^7 + x^6 + x^5 + x^4 + 1) \pmod{x^8 + 1}.$$

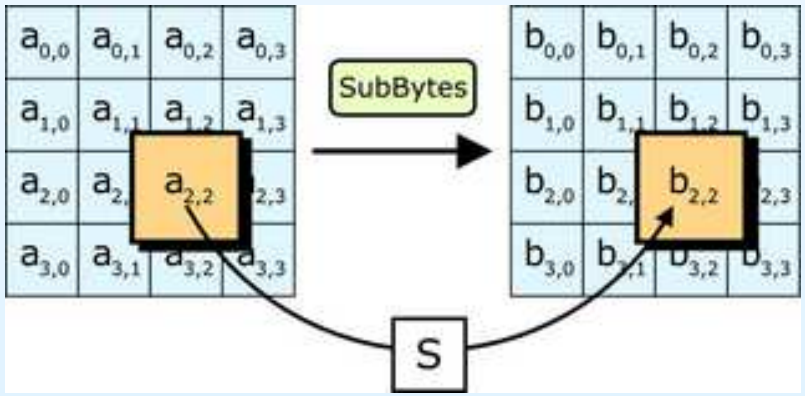
- $\mathcal{S}(x)$ is differentially 4-uniform
- $\mathcal{N}(\mathcal{S}) \geq 2^7 - 2^4 = 112$, which is very close to the upper bound of $2^7 - 2^3 = 120$.
- These properties, combined with the level of diffusion in the linear functions, and the number of rounds, make Rijndael provably secure against linear and differential cryptanalysis.




*The End*




...and there was much rejoicing...